Advanced Algorithmics (6EAP)
Project proposals

Jaak Vilo
2011 Spring
Key info

- Project = 1-2-3 person teams
- Deadline: May 26th
- Delay: 1 day = 20% of max points
- Prerequisite for exam
Expectations:

• Study the problem
• Implement, Evaluate, Compare, Measure, ...
• Your task is to make the project interesting to others: right questions; cool applications; novel ideas; desire to read; materials to complement next year courses.
• Find a clear objective and focus, state it, study it!
• 20-40h
• Report – 3-5 pages of text, including references; illustrations, tables; ...
Tasks

• Here is a list of some proposals

• You can propose your own.

• Or select some on your own
  – from international competitions
    • e.g. IOI (ACM) olympics finals series
    • implementation challenges from DIMACS, etc.
    • etc.
• Compare some alternative algorithms, and try to improve on them
• Take a problem and try to apply algorithmic problems to solve it
• Take an algorithm from the literature, implement and test
• Find cool ways to study/visualise algorithmic ideas presented in the course.
Combinatorial search

- Optimal solution from a (or any) state
- “Discover” the short assembly step algorithms
- Provide solutions
15-puzzle

- Find an optimal solution
15-puzzle

• Solver for 15-puzzle; solve larger ones(?)
  – goal is to find shortest path solutions
  – try heuristics
  – different sizes of the problem

• Comment: In case of a well-studied problem, expectations on project achievements, comparisons, reporting, a good tutorial style, etc are much higher; it’s harder to show your own creativity.
Binary Heap speedup
Bloom filter(s)

• Bloom filter storage of text using de Bruijn graphs (see video)
Graph layout

• Graph layout
  – “Physical Spring model” with some extra added constraints or specialised nodes for stars, cliques, connection strength, etc.

• Create a nr of criteria and try to minimize nr of crossings, area of graph, etc.
Constrained Spring Embedding Layout

• Define certain areas (or lines, etc) that “attract” nodes. Allow graph to “layout” itself dynamically.
Graph layout II

- Layout a (weighted) graph by trying to minimize nr of visual connections. I.e. develop some “flows”. E.g. make wider connections if many connections from a “module” or region to another

*Figure 9: A communication heat map.*
Visit all cities... - physically!

http://cswww.essex.ac.uk/staff/sml/gecco/PTSPComp.html
http://algoval.essex.ac.uk/ptsp/ptsp.html
• 652, 652
• 648, 636
A* path-finder
Seriation

- Serialise matrices
- (2-way)
Biclustering

• Ordering rows and columns to reveal modules/areas of high “coherence”

• Example: A. Tanay, R. Sharan, R. Shamir: Discovering statistically significant biclusters in gene expression data. Bioinformatics 18, Suppl.1, 136-44, 2002
Alizadeh et al., Nature 403:503-11, 2000
Query of OCT4 (POU5F1) (210265_X_AT)

StdDev < 0.29
Query of OCT4 (POU5F1)
(210265_X_AT: 50 top StdDev datasets in query)
Some algorithmic competition

• Test your skills on some algorithmic competition
Finally, 15,000 pages later:

\[
\begin{array}{cccccccccc}
-7 & 260 & 0 \\
7 & -260 & 0 \\
1072 & 1070 & 0 \\
-15 & -14 & -13 & -12 & -11 & -10 & 0 \\
-15 & -14 & -13 & -12 & -11 & 10 & 0 \\
-15 & -14 & -13 & -12 & 11 & -10 & 0 \\
-15 & -14 & -13 & -12 & 11 & 10 & 0 \\
-7 & -6 & -5 & -4 & -3 & -2 & 0 \\
-7 & -6 & -5 & -4 & -3 & 2 & 0 \\
-7 & -6 & -5 & -4 & 3 & -2 & 0 \\
-7 & -6 & -5 & -4 & 3 & 2 & 0 \\
185 & 0 \\
\end{array}
\]

Combinatorial search space of truth assignments: \(2^{50000} \approx 3.160699437 \cdot 10^{15051}\)

Current SAT solvers solve this instance in approx. 1 minute!
Density based clustering
I SPEXS: general algorithm

1. \( S = \text{input sequences (} \|S\| = n \) \\
2. \( e = \text{empty pattern, } e.\text{pos} = \{1,\ldots,n\} \) \\
3. \( \text{enqueue( } \text{order} \ , \ e \) \\

4. \( \text{while } p = \text{dequeue( } \text{order} \) \\
5. \( \text{generate all allowed extensions } p' \text{ of } p \text{ (& } p'.\text{pos} \) \\
6. \( \text{enqueue( } \text{order, } p' \ , \ \text{priority}(p') \) \\
7. \( \text{enqueue( } \text{output, } p' \ , \ \text{fitness}(p') \) \\

8. \( \text{while } p = \text{dequeue( } \text{output} \) \\
9. \( \text{Output } p \)


Applications in bioinformatics:
- Functional elements in proteins (2002: 32 cit)
Sequence patterns: the basis of the SPEXS

- GCAT (4 positions)
- GCATA (3 positions)
- GCATA. (3 positions)
- GCATA.C
• priority: *e.g.* depth-first; breadth-first; best-first; A*, beam search, ...

• fitness: most frequent; max(freq * length); ...
Similarity join

• From two sets of objects
  – find the best matching pairs

• E.g. two photos from slightly changed angle
  – identify “same” points on two images
  – can have applications in 3-D stereo imaging; photo stitching (panoramic photos), etc.
Graph edit distance

• Calculate edit distance between graphs:
  – add/delete nodes, edges
  – match nodes (e.g. by partially matching labels)

  – Q: How similar are two graphs
Graph: SESE

• Identify region(s) of **Single Entry, Single Exit**.

• I.e. find subgraphs that can be “collapsed” and studied further recursively, for example

  • Marlon: Here is a set of test cases you could give to the students who choose the project on "identifying SESE regions". The rar file attached contains a README file that explains the contents of the folder and the format used to encode the process graphs.
Graph: counting features

- Count a nr of times certain features exist in graphs; find “most frequent features”
- E.g. small feedback-triangles; feed-forward triangles, etc.
- Hint: count nr of smallest features; add certain feature and recalculate their frequencies.
- Keep expanding the most frequent features (similar to SPEXS idea)
Graph

• Implement Page Rank algorithm
  – Evaluate problem sizes, provide examples
  – Abstract graphs
  – Method could be used on scientific literature, social networks, etc...
  – Can you add more features, prevent “spammers”?
Your own projects

- Ask a question
- Study literature
- Propose solution
- Implement
- Experiment and report results of experiments